# Инфо

Аппаратные ускорители играют ключевую роль в задачах машинного обучения, ускоряя процессы обучения нейросетей и обработки данных. Давайте разберёмся, чем отличаются различные типы ускорителей, таких как **CPU**, **GPU** и **TPU**, а также что из этого может быть оптимально для задач машинного обучения, особенно для несложного обучения.

### **1. CPU (Central Processing Unit)**

**Центральный процессор** — это основной вычислительный элемент в любом вычислительном устройстве. Он выполняет все общие операции, включая управление задачами, вводом/выводом и операциями с памятью.

**Преимущества CPU**:

* **Универсальность**: Подходит для широкого спектра задач, не только для машинного обучения.
* **Многозадачность**: Идеален для приложений, которые требуют много вычислений на одном потоке или сложных алгоритмов.
* **Гибкость**: Подходит для несложных моделей машинного обучения и задач обработки данных, где параллельность не так критична.

**Недостатки**:

* Для задач глубокого обучения CPU может быть недостаточно быстрым, так как параллельная обработка на нём ограничена числом ядер.

**Пример использования**:

* Хорошо подходит для обучения моделей с малым количеством данных, в том числе для тестирования и разработки.

### **2. GPU (Graphics Processing Unit)**

**Графический процессор** изначально был создан для обработки графики и видеоигр, но со временем его вычислительные мощности стали использоваться для **машинного обучения**. Современные нейросети, особенно глубокие, требуют большого объема матричных операций, которые GPU обрабатывает очень эффективно.

**Преимущества GPU**:

* **Параллельная обработка**: GPU имеет тысячи малых ядер, которые могут одновременно обрабатывать множество операций, что делает их идеальными для обучения нейросетей.
* **Высокая производительность**: Подходит для задач, требующих большого объема вычислений, таких как обучение глубоких нейросетей и обработка больших массивов данных.

**Недостатки**:

* **Большие расходы**: Аренда GPU может быть дорогой, особенно при использовании мощных моделей, таких как V100 или A100.
* **Меньшая гибкость, чем у CPU**: GPU лучше всего подходят для параллельных вычислений, и не все задачи могут быть эффективно распараллелены.

**Пример использования**:

* Идеальны для сложных задач машинного обучения, таких как обучение глубоких нейросетей на больших объемах данных.

### **3. TPU (Tensor Processing Unit)**

**Тензорные процессоры** — это специализированные процессоры от Google, предназначенные для ускорения операций с тензорами, которые являются основой вычислений в нейросетях. TPU оптимизированы для работы с фреймворками, такими как TensorFlow, и могут значительно ускорить обучение моделей машинного обучения, особенно тех, которые используют много матричных операций.

**Преимущества TPU**:

* **Высокая производительность для TensorFlow**: TPU являются лучшим выбором для глубоких нейросетей, обучаемых на TensorFlow.
* **Энергетическая эффективность**: TPU могут выполнить операции с меньшими затратами энергии по сравнению с GPU.
* **Отличная производительность при обучении моделей**: Высокая пропускная способность для матричных операций, что ускоряет обучение моделей.

**Недостатки**:

* **Ограниченная совместимость**: TPU оптимизированы для работы в экосистеме TensorFlow и не всегда хорошо поддерживают другие фреймворки.
* **Не универсальны**: Для задач, которые не используют матричные операции или не интегрированы с TensorFlow, TPU может не подойти.

**Пример использования**:

* Оптимальны для больших моделей глубокого обучения, особенно если используется TensorFlow и требуется высокая скорость обучения.

### **Сравнение предложенных сервисов:**

* **Google Colab**:  
  + **GPU T4** — хорош для задач машинного обучения с умеренной нагрузкой. Это оптимальный вариант для пользователей, которые работают с относительно несложными моделями.
  + **TPU v2-8** — лучше подходит для более сложных моделей, особенно если вы работаете с TensorFlow, так как TPU предлагает отличную производительность в этой экосистеме.
* **Yandex Cloud**:  
  + **c1.4 (4 vCPUs, 32 GB RAM)** — подойдёт для обучения небольших моделей, когда нет потребности в ускорителях.
  + **gt4.1 (1 GPU T4, 4 vCPUs, 16 GB RAM)** — хорош для небольших моделей, где нужно ускорение благодаря GPU. T4 — это хороший выбор для умеренно сложных задач.
  + **g1.1 (1 GPU V100)** — это более мощный вариант для сложных задач, требующих большего объема вычислений, таких как обучение больших нейросетей.
  + **g2.1 (1 GPU A100)** — это более мощный вариант для серьезных задач с большими моделями, которые требуют максимальной производительности.

### **Что выбрать для несложного обучения:**

Для несложного обучения, например, для обучения небольших нейросетей или работы с небольшими наборами данных, **GPU T4** будет отличным выбором, так как это баланс между ценой и производительностью. В вашем случае, **Google Colab с GPU T4** — это удобное и экономичное решение, которое обеспечит нужную производительность для большинства задач.

Если в будущем задачи будут усложняться, можно рассматривать использование **GPU V100 или A100** для большего ускорения или **TPU**, если вы планируете работать с TensorFlow и сложными моделями.

# Google Colab vs Yandex Cloud

Сравнение базовых версий

Google colab

Model name: Intel(R) Xeon(R) CPU @ 2.20GHz

CPU family: 6

Model: 79

Thread(s) per core: 2

Core(s) per socket: 1

Socket(s): 1

Stepping: 0

BogoMIPS: 4399.99

выполнил задание за 38 секунд

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Yandex Cloud

Model name: Intel Xeon Processor (Cascadelake)

CPU family: 6

Model: 85

Thread(s) per core: 2

Core(s) per socket: 2

Socket(s): 1

Stepping: 6

BogoMIPS: 4190.16

выполнил задание за 25 секунд

Из вашего вывода видно, что на двух разных процессорах были выполнены одинаковые задания, но на разных моделях процессоров:

1. **Intel Xeon @ 2.20GHz (модель 79)**: Этот процессор имеет 1 ядро с 2 потоками, и задание выполнилось за 38 секунд.
2. **Intel Xeon Processor (Cascadelake)**: Это более новый процессор с 2 ядрами и 2 потоками на каждом ядре. Он выполнил задание за 25 секунд.

Основное отличие между этими двумя процессорами — это количество ядер. Второй процессор, с 2 ядрами, очевидно, справляется с задачей быстрее, вероятно, за счет более высокой параллельной обработки данных.

Более высокие **BogoMIPS** также могут указывать на более эффективную обработку инструкций, что также сказывается на скорости выполнения задачи.

# Yandex DS: CPU vs GPU

### Задача

def objective(trial):

n\_layers = trial.suggest\_int("n\_layers", 1, 15)

layer\_configs = [trial.suggest\_int(f"layer\_{i+1}\_neurons", 1, 250) for i in range(n\_layers)]

#batch\_size = trial.suggest\_float("batch\_size", 0.005, 0.3)

batch\_size = 0.001

#lr0 = trial.suggest\_float("lr0", 0.01, 1.0)

lr0 = 0.025

#lr1 = trial.suggest\_float("lr1", 0.00001, 0.01)

lr1 = 0.00075

model = DynamicNet(5,layer\_configs,1, device)

loss = pipe2time(model, data, feat=feat, bs=batch\_size, lr0=lr0, lr1=lr1, eps=eps, device=device, print\_loss = print\_loss)

return loss

study = optuna.create\_study(directions=["minimize"])

study.optimize(make\_objective(print\_loss=0, feat = feat, eps = 5), n\_trials=60)

### 

### СPU

c1.4 4 vCPUs, 32 ГБ RAM

Intel Xeon Processor (Cascadelake)

Самая дешевая за 18 руб в час

Model name: Intel Xeon Processor (Cascadelake)

CPU family: 6

Model: 85

Thread(s) per core: 2

Core(s) per socket: 2

Socket(s): 1

Stepping: 6

BogoMIPS: 4190.15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Execution started at 2025-04-13 17:09:14

Last executed at 2025-04-13 18:18:21 in 1h 9m 6s

feature: 5

Модель 1: Слои=6, Loss=1.875973

Конфигурация слоев: [190, 73, 151, 122, 111, 94]

Модель 2: Слои=5, Loss=1.883954

Конфигурация слоев: [188, 74, 80, 137, 23]

Модель 3: Слои=4, Loss=1.901334

Конфигурация слоев: [207, 74, 128, 137]

Модель 4: Слои=3, Loss=1.922877

Конфигурация слоев: [163, 13, 79]

Модель 5: Слои=3, Loss=1.989011

Конфигурация слоев: [196, 19, 59]

### GPU

gt4.1 1 GPU T4, 4 vCPUs, 16 ГБ RAM

Tesla T4

самая дешевая gpu 155 руб в час

╔══════════════════════════════════════════════════════════════════════════════╗

║ 🖥️ NVIDIA-SMI 535.216.03 Driver: 535.216.03 CUDA: 12.2 ║

╠════════════════════════════════════╤══════════════╤═════════════════════════╣

║ GPU Name │ Persistence-M│ Bus-Id Disp.A ║

║────────────────────────────────────┼──────────────┼─────────────────────────║

║ 0 Tesla T4 (On) │ Off │ 00000000:8C:00.0 ║

╠════════════════════════════════════╧══════════════╧═════════════════════════╣

║ 🔥 Temp: 27°C 🌀 Fan: N/A ⚡ Pwr: 9W/70W (P8) ║

║ 💾 RAM: 7MiB/15360MiB 🔧 Util: 0% (Default) ║

╠══════════════════════════════════════════════════════════════════════════════╣

║ 🖥️ Processes: ║

╠══════════════════════════════════════════════════════════════════════════════╣

║ PID 833: /usr/lib/xorg/Xorg (4MiB) ║

╚══════════════════════════════════════════════════════════════════════════════╝

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Execution started at 2025-04-13 17:47:31

Last executed at 2025-04-13 18:25:24 in 37m 52.85s

feature: 5

Модель 1: Слои=3, Loss=1.961099

Конфигурация слоев: [178, 50, 14]

Модель 2: Слои=3, Loss=1.963314

Конфигурация слоев: [177, 48, 9]

Модель 3: Слои=3, Loss=2.031883

Конфигурация слоев: [168, 48, 7]

Модель 4: Слои=3, Loss=2.067994

Конфигурация слоев: [167, 49, 5]

Модель 5: Слои=5, Loss=2.144033

Конфигурация слоев: [187, 102, 19, 177, 164]